IN-17
185026
10 P

NASA Technical Memorandum 105879

SCAILET: An Intelligent Assistant for Satellite Ground Terminal Operations

A.K. Shahidi, J.A. Crapo, and R.F. Schlegelmilch
Sverdrup Technology, Inc.
Lewis Research Center Group
Brook Park, Ohio

R.C. Reinhart
Analex Corporation
Brook Park, Ohio

and

E.J. Petrik, J.L. Walters, and R.E. Jones
National Aeronautics and Space Admnistration
Lewis Research Center
Cleveland, Ohio

Prepared for
The World Space Congress
cosponsored by the AI AA, NASA, and NAS
Washington, D.C., August 28–September 5, 1992

NASA

SCAILET: AN INTELLIGENT ASSISTANT FOR SATELLITE GROUND TERMINAL OPERATIONS

A.K. Shahidi, J.A. Crapo, and R.F. Schlegelmilch
Sverdrup Technology, Inc.
Lewis Research Center Group
Brook Park, Ohio 44142

R.C. Reinhart
Analex Corporation
3001 Aerospace Parkway
Brook Park, Ohio 44142

E.J. Petrik, J.L. Walters, and R.E. Jones
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135

Abstract

NASA Lewis Research Center has applied artificial intelligence to an advanced ground terminal. This software application is being deployed as an experimenter interface to the link evaluation terminal (LET) and has been named Space Communication Artificial Intelligence for the Link Evaluation Terminal (SCAILET). The high-burst-rate (HBR) LET provides 30-GHz-transmitting and 20-GHz-receiving, 220-Mbps capability for wideband communications technology experiments with the Advanced Communication Technology Satellite (ACTS).

The HBR-LET terminal consists of seven major subsystems. A minicomputer controls and monitors these subsystems through an IEEE-488 or RS-232 protocol interface. Programming scripts (test procedures defined by design engineers) configure the HBR-LET and permit data acquisition. However, the scripts are difficult to use, require a steep learning curve, are cryptic, and are hard to maintain. This discourages experimenters from utilizing the full capabilities of the HBR-LET system. An intelligent assistant module was developed as part of the SCAILET software. The intelligent assistant addresses critical experimenter needs by solving and resolving problems that are encountered during the configuring of the HBR-LET system. The intelligent assistant is a graphical user interface with an expert system running in the background. In order to further assist and familiarize an experimenter, an on-line hypertext documentation module was developed and included in the SCAILET software.

1.0 Introduction

System studies performed during the late 1970's indicated that advanced communications satellite technologies should be developed to utilize the Ka-band (30/20 GHz) spectrum. In order to demonstrate Ka-band satellite communications systems, NASA Lewis Research Center has conducted an advanced space communications program. This program will meet the needs of future NASA missions and infuse advanced technologies into the commercial sector. As a result, a number of Ka-band satellite communications architectures and their associated technologies were studied. In order to demonstrate and validate Ka-band satellite communications with advanced technologies such as multibeam antennas, baseband processing, satellite microwave switch matrix, advanced modulation and coding, and adaptive signal fade compensation, NASA Lewis began developing an Advanced Communications Technology Satellite (ACTS).

In addition, a high-burst-rate link evaluation ground terminal (HBR-LET) is being developed as a key part of the ACTS ground-segment experimenter program. The HBR-LET is a versatile and adaptable ground terminal. It was developed to characterize the on-orbit performance of ACTS. It provides reception, transmission, and switch matrix control functions, along with modulation/demodulation processing. The HBR-LET will uplink/downlink at data rates of up to 220-Mbps. It will allow the radio frequency (RF) characterization of the satellite to be performed. It will also be used to demonstrate adaptive uplink power control for signal fades in a continuously variable manner by employing an advanced signal fade prediction algorithm.

Artificial intelligence (AI) techniques are being developed to enable autonomy and fault management of the HBR-LET. These AI techniques are being incorporated into the ground terminal to facilitate the maintenance, use, training, and documentation of the ground terminal hardware and software systems. Traditionally, the same requirements have been provided by paper manuals, experimenter guides, classroom sessions, or blueprints and other hard-copy material. The AI applications will allow these requirements to be entirely computerized. Additionally, the computer gives the ability to include some functionality that is not possible with traditional methods.

1.1 HBR-LET Subsystems

The HBR-LET consists of seven major subsystems:

(1) Antenna subsystem
(2) RF transmitter subsystem
(3) RF receiver subsystem
(4) Controlling and performance monitoring (C&PM)subsystem
(5) Local loopback subsystem at RF
(6) Modulation and bit error-rate (BER) measurement subsystem
(7) Calibration subsystem

The C&PM subsystem controls and monitors all other subsystems through an IEEE-488 or RS-232 protocol interface. HBR-LET experiments with the ACTS are initiated by experimenters through the C&PM experiment controlling and monitoring (EC&M) software. The EC&M provides experimenters with the ability to control instrumentation used in HBR-LET experiments with the ACTS satellite. The EC&M software was developed on a Concurrent Computer Corp. 3205 minicomputer in FORTRAN.

1.2 HBR-LET Ground Terminal

The HBR-LET ground terminal will be available to industry, university, and government agencies for conducting experiments associated with the ACTS. Tests are planned to characterize the communications link by performing bit error rate[1] versus E_b/N_0[2] tests using various terminal configurations. BER versus E_b/N_0 tests are a popular means of characterizing the performance of a communications system.

Uplink power control is performed by using a signal fade prediction algorithm to determine when additional power is required to compensate for a signal fade event. The signal fade prediction algorithm uses beacon data from the ACTS. The data are analyzed by the beacon measurement subsystem that was developed to monitor the real-time signal fade. The algorithm will predict future signal fades and augment the uplink power to sustain a specified BER. Once the fade perturbation is removed, the uplink power is returned to a level suitable for clear sky.

The C&PM subsystem assists experimenters in performing experiments using the HBR-LET. The subsystem software consists of several software applications to control the HBR-LET instrumentation, to provide the ability to program the microwave switch matrix onboard the ACTS, to augment the HBR-LET uplink power, and to display acquired data please reference [1].

2.0 SCAILET

It became clear early in the project that configuring the HBR-LET would be difficult for experimenters. Because experimenters will be coming from a variety of backgrounds (e.g., academia, government, and industry), they needed better tools for designing experiments. Section 2.1 discusses the problems of the original HBR-LET FORTRAN program, section 2.2 discusses the intelligent assistant module, and section 3.0 describes the hypertext documentation module.

2.1 Original Controlling and Performance Monitoring Software

A major aspect of the C&PM software allows an experimenter to control and monitor the instrumentation used in the HBR-LET experiments by using the EC&M software. The EC&M software interface is a menu driver that is displayed on an ASCII terminal. The menu driver is slow and does not allow experimenters to move around the different screens easily. The HBR-LET instrumentation is controlled through predefined test sequences that are executed by the HBR-LET minicomputer. HBR-LET experimenters develop the test sequences according to their specific requirements. The minicomputer controls the instrumentation through a communications interface.

In order to configure the HBR-LET for an experiment, an experimenter must choose and initialize a set of instruments. They are chosen from a pool of available instruments. The process of choosing instruments creates an *instrument definition file*. Next, the experimenter programs the instruments by using a set of commands called sequences. This process generates a *sequence definition file.* The experimenter can then specify the associative file names. These files are created by the instrument definition and sequence definition software. This software created a controlled, restricted, and unforgiving environment for developing instrument and sequence definition files. Moreover, editing or changing these existing files can be difficult for experimenters and requires learning additional software.

The user interface for developing the instrument and sequence definition files was an ASCII terminal that took the experimenter through a series of menu-driven text screens. It was this interface and the lack of editing capabilities that made programming the HBR-LET cumbersome. This section describes the difficulties of the interface for each of the processes.

At the beginning of the instrument definition process the experimenter needed to have a mental model of the HBR-LET circuitry and all its possible paths (i.e., loopback with traveling-wave tube amplifier (TWTA) mode, loopback without the TWTA mode, the calibration mode, and the LET-ACTS-LET mode). Then the experimenter had to reference which instruments would be available to be monitored. This would be done by referencing documentation that is associated with each subsystem in the HBR-LET circuitry path. A first time experimenter also had to reference the description of each instrument within the subsystem in order to understand what it monitored, and to decide whether it would be necessary for the experiment. Up until this point the experimenter had not even touched the keyboard of the computer. Much time had been spent investigating difficult details rather than gearing energies toward the experiment and results. This would make experimenting difficult for new experimenters.

After reviewing instrument-related details the experimenter could proceed to an instrument selection screen where all available instruments corresponding to the HBR-LET would be listed. Figure 1 illustrates the software system Instrument Table. Experimenters could select instruments from those available by using keyboard commands. Instruments preceded by an X represent those selected by the experimenter. Column headings represent the instrument label, the NASA identification number, and the instrument identification mnemonic. However, if the experimenter made a mistake by choosing an instrument that was not in the path or did not need to be monitored, the resulting instrument file would have to be redeveloped.

HBR-LET ECAM Software System Instrument Table

X	NUM	LABEL	TAG	ID	X	NUM	LABEL	TAG	ID	X	NUM	LABEL	TAG	ID
X	01	PM1	[illegible]	WTPM		02	B37	G50340	HPSO	X	03	N20	071640	HPPM
X	04	L16	071646	HPPM	X	05	PIN	071580	HPPS		06	U13	[illegible]	HPAT
X	07	R12	[illegible]	HPAT	X	08	C7	071536	HPRA	X	09	C9	071567	EPF
X	10	DG1		DG	X	11	DC1		DC	X	12	DGT		DGT

Figure 1 HBR-LET Software System Instrument Table.

After selecting the experiment instruments, the experimenter would proceed to view each instrument's initialization screen one by one. Default values would be presented with the option to modify. Menu choices would be presented in a cryptic format. For instance, a power meter has as option to define an *out-of-limit action.* The possible choices associated with this operation are *halt a test, continue,* or *other* (experimenter-defined actions). Actual entry would require the experimenter to enter 0,1,2 respectively. The definition of each number is presented in a separate documentation box, but the process and the entry are not intuitive. Once the experimenter had initialized all the instruments, the instrument definition file would be generated. Each value that was associated with the initialization of the instrument would be sequentially written in a 40-position array (Figure 2). In cases where the experimenter wanted to modify an existing instrument file, the array file would need to be modified. This would mean referencing an additional manual that describes the functionality of each position in the array. Using an editor, the experimenter would have to edit each incorrect array position.

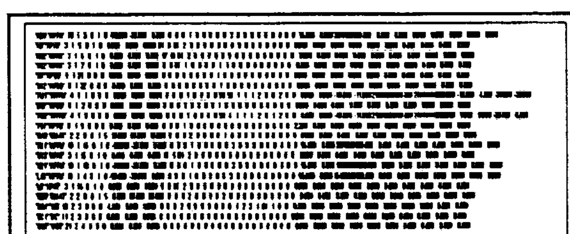

Figure 2 A typical instrument definition file for 20 instruments.

The sequence definition process was not any easier. Figure 3 shows the list of commands available to the experimenter for developing scripts. As with the instrument definition, scripts were created by

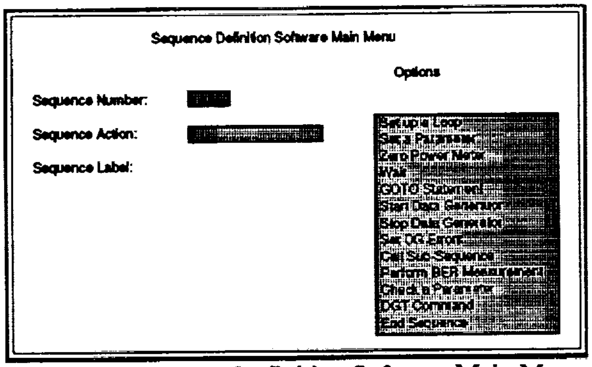

Figure 3 Sequence Definition Software Main Menu.

using keyboard commands. The experimenter had to go through a set of menu-driven screens searching for the proper command. Each command had its own menu-driven screen. The resulting sequence file was a sequential 24-position array for each command (Figure 4). The file was cryptic and difficult to understand. Editing this file also required a separate manual and text edition.

```
100000 0 0 101 2 2 0 0 0 'WAIT' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
300000 0 0 0 0 0 0 0 0 'CSUB' 0 0 'M1Z' 'ERO' '90 '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 21 6 0 0 0 0 'TGSW' 0 0' '' '' '' '' '0 0 0 0 0 0 '100000000'
100000 1 0 301 18 0 0 0 0 'STDG' 1 0' '' '' '' '' '8 15 0 1 0 0 '000000000'
100000 0 0 110 14 0 0 -50 -20 'CKPR' 100999 100999' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 21 13 0 0 14 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100030 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100010 0 0 61 13 0 0 -1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100030 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100020 0 0 61 13 0 0 1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100030 0 0 110 7 0 0 -15 -13.5 'CKPR' 100010 100020' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100090 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100070 0 0 61 16 0 0 -1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100090 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100080 0 0 61 16 0 0 1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100090 0 0 110 15 0 0 -46 -44 'CKPR' 100070 100080' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100120 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100100 0 0 61 3 0 0 -1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 102 0 0 0 0 0 'GOTO' 100120 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100110 0 0 61 3 0 0 1.0 0 'ATTN' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100120 0 0 110 1 0 0 -39 -37.5 'CKPR' 100100 100110' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 21 6 0 0 0 0 'TGSW' 0 0' '' '' '' '' '0 0 0 0 0 0 '110100000'
100000 1 0 302 18 0 0 0 0 'SPDG' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
300000 0 0 0 0 0 0 0 0 'CSUB' 0 0 'M1:D' '110M' '110' '90 '' '0 0 0 0 0 0 '000000000'
100000 0 0 101 6 2 0 0 0 'WAIT' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
300000 0 0 0 0 0 0 0 0 'CSUB' 0 0 'M1:D' '100M' '110' '90 '' '0 0 0 0 0 0 '000000000'
100999 0 0 21 10 0 0 5 0 'PWVT' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
100000 0 0 101 60 10 0 0 0 'WAIT' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
999997 0 0 0 0 0 0 0 0 'ENDM' 0 0' '' '' '' '' '0 0 0 0 0 0 '000000000'
```

Figure 4 A typical sequence definition file with 28 commands.

2.2 SCAILET Intelligent Assistant (SCAILET IA)

The intelligent assistant was developed as a front-end/graphical user interface (GUI) to a minicomputer. It assists the experimenter in generating and modifying instrument definition and sequence files. Figure 5 exhibits the architecture of the intelligent assistant, the minicomputer, and the HBR-LET.

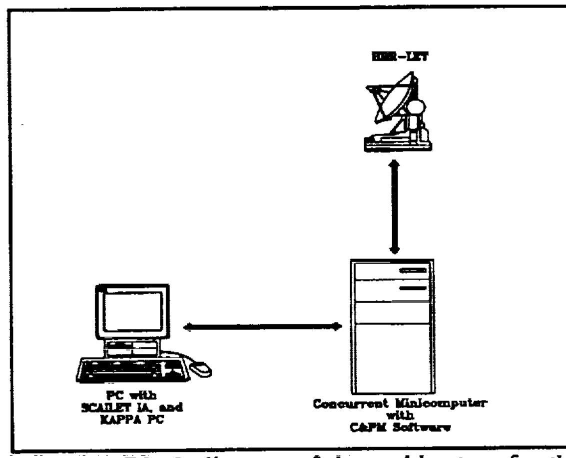

Figure 5 Block diagram of the architecture for the HBR-LET.

SCAILET IA resides on a DOS-based personal computer with a 80486DX processor. It was developed by using the *ToolBook*[3] Windows Application Development Environment. The *KAPPA PC*[4] expert system shell runs in the background and communicates with ToolBook through the Dynamic Data Exchange (DDE) Protocol. SCAILET IA is linked to the minicomputer by a custom communications link that utilizes the RS-232 protocol. The output of the SCAILET IA is identical to that produced by the C&PM software (i.e., instrument definition file of Figure 2, and sequence definition file of Figure 4). However, the experimenter is completely unaware of the cryptic commands being generated because the GUI shields the complexity.

The first task of SCAILET IA was to reduce the initial work load that had to be done by the experimenter before programming an experiment. SCAILET IA simulates a complete "live" block diagram of the HBR-LET system, enabling the experimenter to configure and observe any of the possible configuration modes. The experimenter can choose any mode by pointing and clicking on any of the switches that connect the subsystems. If the experimenter has questions regarding any subsystem, on-line hypertext documentation can be activated to provide descriptions and experimental documentation that are associated with the HBR-LET. (The hypertext documentation module is described in detail in section 2.3.) Upon choosing the desired path, a list of the instruments that are available to be chosen is provided. Hypertext documentation is available for each instrument. After understanding the functionality of each instrument, the experimenter can choose to use or discard any available instruments. Moreover, the experimenter can choose to modify the default values of any selected instrument. Choices are no longer numeric as they were on the ASCII terminal of the EC&M software interface; rather they are dynamic pulldown menu selections. The experimenter is able to backtrack at any point in order to correct mistakes. Corrections can be performed without editing cryptic files or restarting from scratch.

Figure 6 shows a typical instrument initialization screen. The complex instrument definition file is

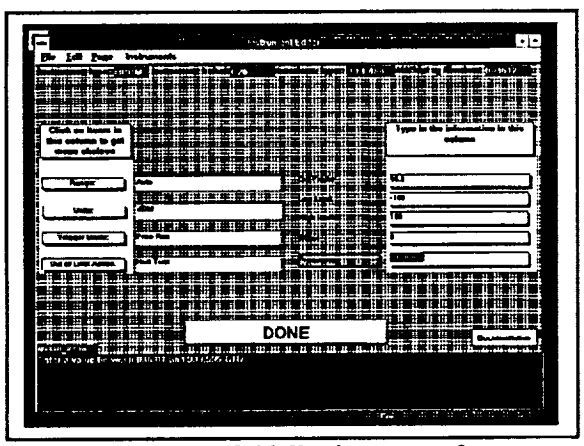

Figure 6 Instrument initialization screen for a power meter.

created, but it is transparent to the experimenter. The experimenter sees only the GUI and the actual values. The power of this system comes in editing existing instrument definition file. The experimenter need only choose a predefined instrument file to edit. The file is then opened and the experimenter can add any additional instruments or modify any instrument values and save it as a separate file name. The editing process is no longer a tedious and cryptic one.

The sequence definition process has also been simplified a great deal. A constraint language with its own syntax has been developed. It produces the sequence definition commands like any other programming language but is extremely user friendly. For example, the old syntax for zeroing a power meter used to be as shown in Figure 7. The command specifies the action, the type of instrument, and the specific instrument. SCAILET IA sends this

```
10000000060 10000'ZERO'00' '' ''   '' '' '000000'000000000'
```

Figure 7 ASCII syntax used to zero a power meter.

command by selecting the menu choice exhibited in Figure 8. The experimenter does not have to know the abstract, cryptic command. Furthermore, pulldown menus are used to generate the possible

```
ZERO POWER METER HPPM N20
```

Figure 8 SCAILET IA command for zeroing a power meter.

commands. Commands can only be used for instruments that have already been chosen in the corresponding instrument file. The sequence editor checks the syntax by verifying the experimenter's actions and provides error statements to syntaxually incorrect programs. The experienced experimenter can forego the menu-driven syntax generation and just type the command in the editor. Previously defined sequence files can be recalled and edited by using new syntax. Figure 9 shows the sequence generation environment.

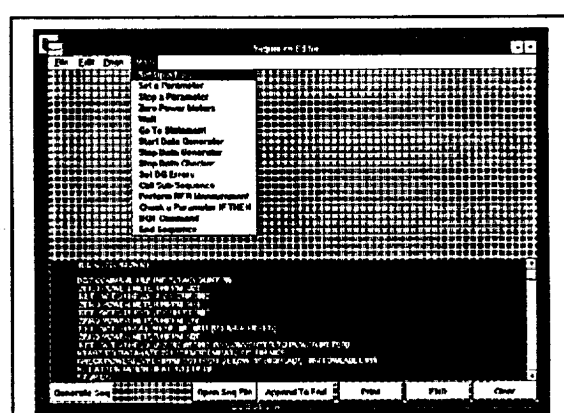

Figure 9 Sequence editor provided in SCAILET IA.

Recalling previously generated files enables the experimenter to recreate the experiment without remembering every detail of the previous program.

3.0 Hypertext Documentation

The HBR-LET is complex. A clear and concise set of documentation is required in order to understand the HBR-LET. Moreover, the experimenters will differ in their satellite communications experience. This further augments the need for documentation that is easily accessible and concisely written.

The HBR-LET *design specifications* were intended to be the only documentation that would be available for experimenters. A set of guidelines was prepared for creating these documents, but the effort had limited success. As a result, the format of the individual subsystem documents differed. By preparing a single reference manual for the HBR-LET, clear and consistent information can be presented to the experimenter.

3.1 What Is Hypertext?

The term *hypertext* was coined back in 1965 by Ted Nelson. Hypertext is defined as nonlinear or nonsequential text. That is, the text is organized so that you can easily move between topics reference [2]. Information is organized into nodes; a node is a small collection of data organized around a specific topic. Early hypertext systems comprised textual nodes only. "Now nodes can contain various kinds of data; graphics, audio, video, computer-animated images, film clips of animated scenes, digital sound or other kinds of information" reference [2].

3.2 Applying Hypertext to HBR-LET Documentation

The "golden" rules of hypertext as described by reference [3] implies that hypertext can be appropriately applied to a collection of information under the following conditions:

(1) There is a large body of information that is organized into numerous fragments.
(2) The fragments relate to each other.
(3) The reader needs only a small fraction at any time.

The HBR-LET design documentation adheres to these rules. The information is organized into numerous fragments; namely the individual HBR-LET subsystem descriptions. These individual fragments are related by design. Also, an experimenter will require only a small fraction of information at any given time.

A hypertext HBR-LET reference manual is being developed. The intent of this manual is to provide an on-line resource for the experimenter. By applying hypertext, experimenters can access information quickly, without being overwhelmed by the nonuniform and vast amount of the HBR-LET design documentation.

3.3 Hypertext Development

The experimenter is given three methods of accessing the subsystem reference documentation. The methods discussed include:

(1) An Instrument Help File
(2) HBR-LET Overview Document
(3) Intelligent Assistant

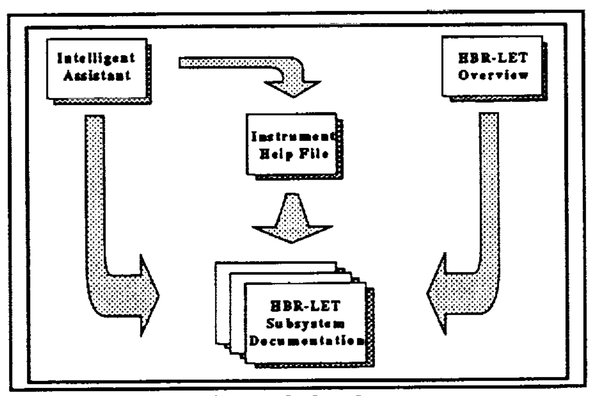

Figure 10 Flow chart relating hypertext documentation to the Intelligent Assistant.

3.3.1 Instrument Help File

The help file is provided as a quick source of information. A situation may arise when additional information about a particular instrument is required. The instrument help file allows the experimenter to access a brief description of the instrument that includes the instrument name and type, the subsystem location, and a monitored or controlled value. The experimenter can double-click on a subsystem for the corresponding documentation. Figure 11 shows the help file.

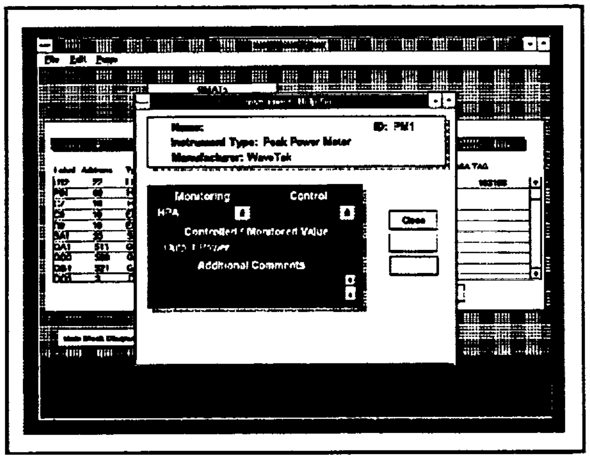

Figure 11 Hypertext help display for the Instrument Editor.

3.3.2 HBR-LET Overview Document

The HBR-LET overview document allows an experimenter to access the documentation independent of the intelligent assistant. Additional information about the HBR-LET and the ACTS Project is provided. As the experimenter pursues the overview, several opportunities are provided to have detailed subsystem information presented.

3.3.3 Intelligent Assistant Interface

The initial screen of the instrument editor also provides access to the individual subsystem documentation modules. This interface is provided to assist the experimenter in selecting the appropriate HBR-LET configuration. By selecting a menu item from the documentation menu or a subsystem from the block diagram, the appropriate documentation is shown.

3.4 Subsystem Documentation Development

One of the fundamental problems with readers of hypertext is the concept of "lost in hyperspace." This phenomenon occurs when the reader becomes entangled in the hypertext web of nodes and cannot systematically establish a reference point. This disorientation can be minimized by applying a structure to the hypertext web.

The structure chosen for this application is a table of contents. Much like the front of a book, the table of contents provides a reference to various nodes within the document. By moving the mouse pointer to a topic and pressing a hot-key you can move directly to the appropriate node.

A reader of a structured hypertext web may still become disoriented. By developing additional navigational controls, the reader will feel more at ease. Navigational controls developed for this project include controls for browsing, backtracking, and establishing a reference point. The browse buttons create the effect of flipping through the pages of a book. The experimenter can step back through previously visited pages by using the backtrack button. A home button or reference button displays the main table of contents for the current subsystem.

3.5 Hypertext Summary

As stated, the HBR-LET is complex. Clear and consistently written documentation is necessary to meet the needs of the various experimenters. The on-line HBR-LET reference manual satisfies these requirements. Accessing this information became more dynamic by applying hypertext techniques. By having an alternative, on-line means of accessing the reference manual, the experimenter can be assured that the answer to a question is only a "click" away.

4.0 Conclusion

The ASCII terminal of the EC&M software was a restrictive interface to experimenters. SCAILET provides experimenters with a graphical user interface that enables them to configure instrumentation, program sequences, and reference documentation. The simplicity associated with data entry, file editing, and on-line assistance makes SCAILET a superior interface to the ASCII terminal. Additionally, with continuous monitoring by the embedded artificial intelligence, the experimenter can be assured of conducting nearly flawless configuration and execution of an HBR-LET experiment.

5.0 Future Direction

Computer-assisted instruction (CAI) is a traditional computer-based training program that takes the experimenter through a predetermined set of lessons. The advent of artificial intelligence technology and advances in cognitive psychology gave rise to intelligent tutoring systems (ITS) as an improvement to CAI. In an ITS environment the curriculum designer determines what concepts the student should learn in a lesson. The student is taken through subjects to see which concepts he or she is lacking.

The program then determines the curriculum that fits the needs of the student.

Concepts important to the operation of each HBR-LET subsystem will be identified, and then a guided learning process will be developed to provide ITS instructions in SCAILET on the operation of the HBR-LET system.

Disclaimer

Trade names or manufactures' names are used in this report for identification only. This usage does not constitute an official endorsement, either expressed or implied, by the National Aeronautics and Space Administration.

End Notes

1. Number of bits in error out of the total number of bits received,or BER.

2. Ratio of energy per bit to noise power density.

3. ToolBook is a registered trademark of Asymetrix Corporation, Bellevue, WA.

4. KAPPA PC is a registered trademark of IntelliCorp, Inc. Mountain View, CA

References

1. A Software Control System for the ACTS High Burst Rate Link Evaluation Terminal, NASA Technical Memorandum 105207, R. Reinhart and E. Daugherty, December 1991.

2. Seyer, Philip, Understanding hypertext: concepts and applications, Windcrest Books, Blue Ridge Summit, PA , 1991, ISBN 0-8306-9108-1

3. Shneiderman, Ben, Kreitzberg, Charles, Berk, Emily, "Editing to Structure a Reader's Experience," Hypertext. Hypermedia Handbook, Emily Berk and Joseph Devlin; Editors, McGraw-Hill Publishing, New York, New York, 1991, pp143-64

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1993	Technical Memorandum

4. TITLE AND SUBTITLE

SCAILET: An Intelligent Assistant for Satellite Ground Terminal Operations

5. FUNDING NUMBERS

WU–506–72–1D

6. AUTHOR(S)

A.K. Shahidi, J.A. Crapo, R.F. Schlegelmilch, R.C. Reinhart, E.J. Petrik, J.L. Walters, and R.E. Jones

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–7302

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM–105879

11. SUPPLEMENTARY NOTES

Prepared for The World Space Congress cosponsored by the AIAA, NASA, and NAS, Washington, D.C., August 28–Septemeber 5, 1992. A.K. Shahidi, J.A. Crapo, and R.F. Schlegelmilch, Sverdrup Technology, Inc., Lewis Research Center Group, 2001 Aerospace Parkway, Brook Park, Ohio 44142; R.C. Reinhart Analex Corporation, 3001 Aewrospace Parkway, Brook Park, Ohio 44142; E.J. Petrik, J.L. Walters, and R.E. Jones, NASA Lewis Research Center, Cleveland, Ohio. Responsible person, R.F Schlegelmilch, (216) 433–3428.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 17

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

NASA Lewis Research Center has applied artificial intelligence to an advanced ground terminal. This software application is being deployed as an experimenter interface to the link evaluation terminal (LET) and has been named Space Communication Artificial Intelligence for the Link Evaluation Terminal (SCAILET). The high-burst-rate (HBR) LET provides 30-GHz-transmitting and 20-GHz-receiving, 220-Mbps capability for wideband communications technology experiments with the Advanced Communication Technology Satellite (ACTS). The HBR-LET terminal consists of seven major subsystems. A minicomputer controls and monitors these subsystems through an IEEE-488 or RS-232 protocol interface. Programming scripts (test procedures defined by design engineers) configure the HBR-LET and permit data acquisition. However, the scripts are difficult to use, require a steep learning curve, are cryptic, and are hard to maintain. This discourages experimenters from utilizing the full capabilities of the HBR-LET system. An intelligent assistant module was developed as part of the SCAILET software. The intelligent assistant addresses critical experimenter needs by solving and resolving problems that are encountered during the configuring of the HBR-LET system. The intelligent assistant is a graphical user interface with an expert system running in the background. In order to further assist and familiarize an experimenter, an on-line hypertext documentation module was developed and included in the SCAILET software.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Space research; Astronautics	10
	16. PRICE CODE
	A02

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102